Decomposition Approach of Matrix Computations

LU Decomposition

LU Decomposition with Partial Pivot

LU Decomposition with Complete Pivot

Monica Bernard
COT 5405
PID: 4535013
Group 14

What are you trying to do?

- We are learning about LU decomposition, partial pivoted LU decomposition and complete pivoted LU decomposition.
- In the process, we will be implementing open source code(s) that perform the decomposition.
- We want to implement the source code that is optimized with regards to memory usage and time consumption.

How is it done today?

- For LU decomposition without pivoting on input matrix A, we find the L and U, which are lower and upper triangular matrices using row echelon method and verify A = LU.
- For partially pivoted LU decomposition, we find a permutation matrix P which is used to reorder the rows of the input matrix A based on the largest value of the pivot column. Then we perform the normal LU decomposition to find the lower and upper triangular matrices L and U. We verify if PA = LU.
- Complete pivoting of LU decomposition as mentioned in the paper is $P^TAQ = LU$, where:

  A is the given input square matrix.

  P and Q are permutation matrices that shift rows and columns.

  L and U are lower and upper triangular matrices respectively.

  - First, we have to make sure that the largest value in the input matrix ‘A’ is brought to position (1,1) by multiplying the input matrix with the permutation matrices. Then a normal LU decomposition is performed to get L and U.
  - After the first LU decomposition is done, the next largest value in the matrix is brought to position (2,2) by multiplying the next set of permutation matrices. Again another LU decomposition is performed to get the next L and U.
  - Based on the size of the input matrix, this process is repeated until the largest values in the matrix at each step are the diagonal elements.
  - We take the transpose of the permutation matrix P and substitute all the matrices in the formula to verify if the right hand side of the equation is equal to the left hand side.

What are the limitations of the current practice?

- Based on what the paper mentions for the process of complete pivoting of LU decomposition, we have to take the transpose of the permutation matrix P where matrix P is used to reorder the rows of the input matrix at each step.
- We think that the decomposition can be performed the same way; that is, by following all the required steps for the decomposition, except that we can achieve the final answer without the necessity to consider transpose of P.

What is new in your approach?

- Our approach is to perform the complete LU decomposition without having the need to take a transpose of the permutation matrix P.
  - *The formula that we will follow in our implementation is PAQ = LU. Instead of, $P^TAQ = LU$ as mentioned in the paper.*
  - *We will be proving that skipping this step still allows us to reach the correct factorization.*
- We also aim at writing an open source application to optimize different parameters like the execution time and memory consumption of the implementation and the number of code lines.
- We also want to show that partial pivoting is a good compromise between accuracy and resources as opposed to complete pivoting.

Why do you think it will be successful?

- We know for a fact that the improvement to complete pivoting has one less step as opposed to the one suggested in the paper.
- This will reduce the arithmetic cost of the solution by reducing the number of code lines, execution time and memory required.
- One note to consider is, that while partial pivoting is less of a computational strain it does have limitations.
  - *Partial pivoting does not work well with rank deficient matrices because it only searches a single column for a pivot and is not able to determine that all remaining potential pivots are zero.*
  - *However in practice, encountering a rank deficient matrix in numerical algebra is extremely rare.*
  - *Due to this, partial pivoting is a good compromise between accuracy and computation complexity.*

Audience benefiting from the new approach.

- Computers usually solve square systems of linear equations using the LU decomposition and it is also a key step when inverting a matrix or computing the determinant of a matrix.

- By making our application an open source code, we hope to make it more available for mathematical enthusiasts who study matrix computations and computer engineers who work closely to understand how computers solve systems of equations.

Risks?

- The skipping the transpose step of the complete pivot LU decomposition might not have as huge of an improvement over the paper's method for complete LU decomposition.

- The reduction in complexity from complete to partial pivot LU decomposition might not be significant enough to justify it.
  - Complete pivoting with LU decomposition has the ability to detect rank deficient matrices and thus if our implementation does not have significantly lower execution time and memory usage, the loss of stability in partial pivoting is not a good compromise.

Mid-term and final steps of the project proposal

- As mid-term implementation/testing, we would like to implement an open source application that can do LU decomposition without any pivoting (A = LU) and LU decomposition with partial pivoting (PA = LU).

  (NOTE: LU decomposition with partial pivoting involves only one permutation matrix P that reorders only the rows of the input matrix to achieve decomposition)

- For final implementation/testing, we want to build an open source application that can do LU decomposition with complete pivoting (PAQ = LU).
- The implementation and concepts of LU decomposition and partially pivoted LU decomposition from mid-term will be used and built upon in the final implementation when finding completely pivoted LU decomposition.
- We will also be checking for benchmarks for 10 different matrix decompositions in MATLAB. We will be checking for runtime, memory usage and the correctness of the decomposition.